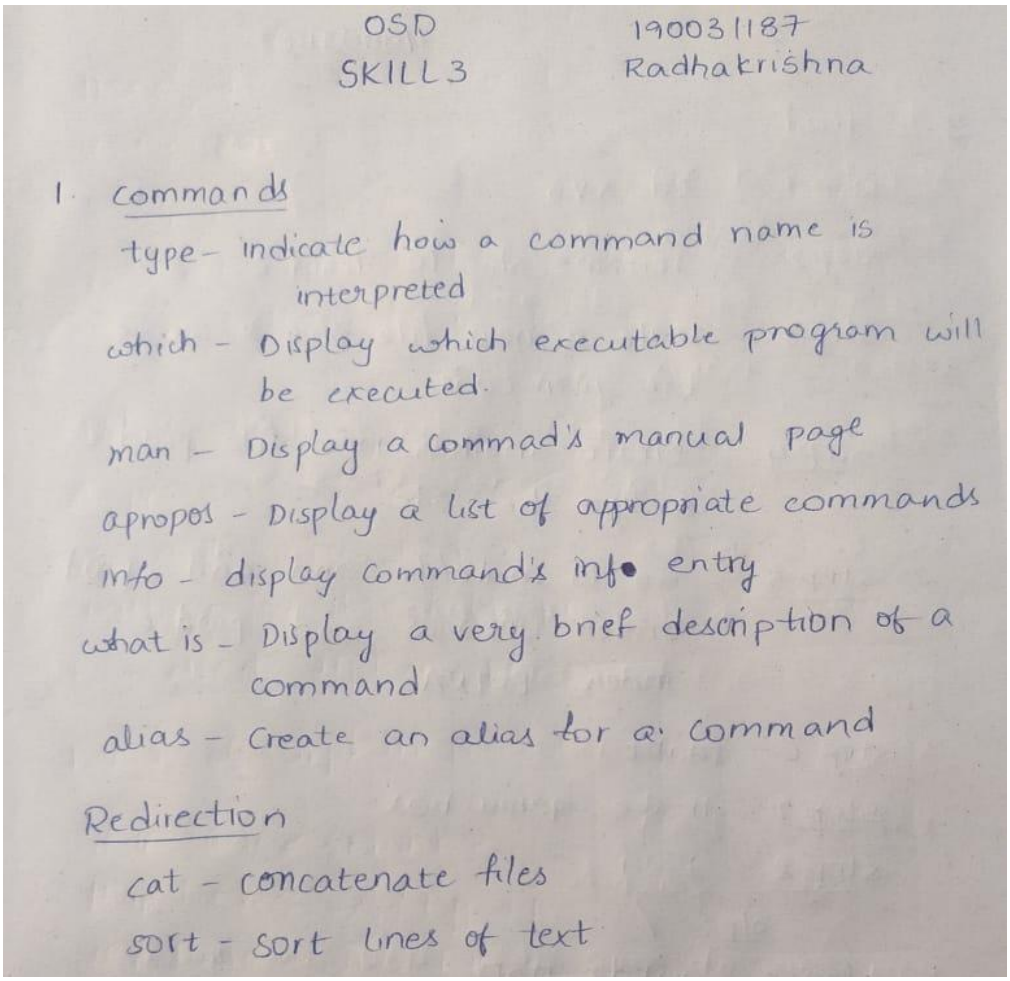
Operating System Design – 19CS2106S

Skill Experiment – 3

1. **Working with Commands, Redirection**



OSD 190031187
SKILL 3 Radhakrishna

1. Commands

type - Indicate how a command name is interpreted

which - Display which executable program will be executed.

man - Display a commad's manual page

apropos - Display a list of appropriate commands

info - display command's info entry

what is - Display a very brief description of a command

alias - Create an alias for a command

Redirection

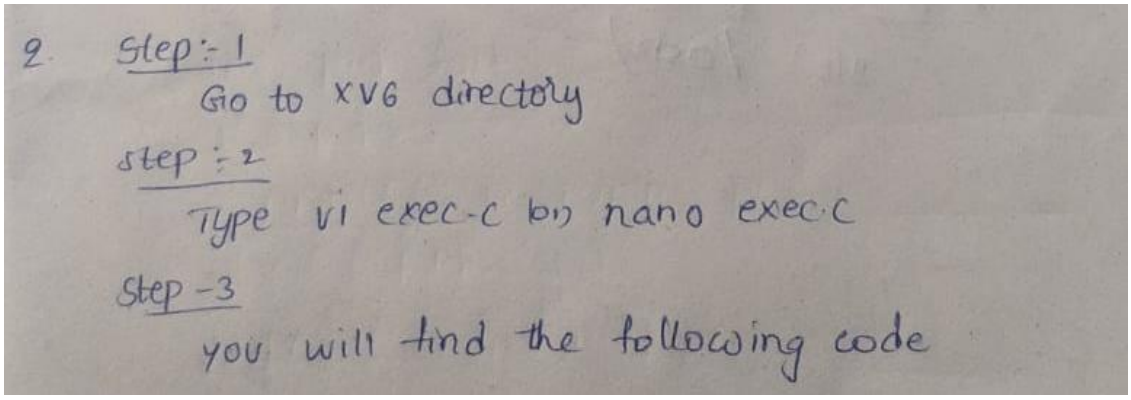
cat - concatenate files

sort - Sort lines of text

osd-190031187@team-osd:~

```
[osd-190031187@team-osd ~]$ type ls
ls is aliased to `ls --color=auto'
[osd-190031187@team-osd ~]$ type cp
cp is /usr/bin/cp
[osd-190031187@team-osd ~]$ which ls
alias ls='ls --color=auto'
        /usr/bin/ls
[osd-190031187@team-osd ~]$ man ls
[osd-190031187@team-osd ~]$ apropos floppy
fd (4)               - floppy disk device
fdformat (8)         - low-level format a floppy disk
mbadblocks (1)       - tests a floppy disk, and marks the bad blocks in the FAT
mformat (1)          - add an MSDOS filesystem to a low-level formatted flopp...
[osd-190031187@team-osd ~]$ whatis ls
ls (1)               - list directory contents
ls (1p)              - list directory contents
[osd-190031187@team-osd ~]$ info coreutils
[osd-190031187@team-osd ~]$ alias
alias egrep='egrep --color=auto'
alias fgrep='fgrep --color=auto'
alias grep='grep --color=auto'
alias l.='ls -d .* --color=auto'
alias ll='ls -l --color=auto'
alias ls='ls --color=auto'
alias vi='vim'
alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot --show-ti
lde'
[osd-190031187@team-osd ~]$
```

2. **exec.c (Xv6 design & implementation)**



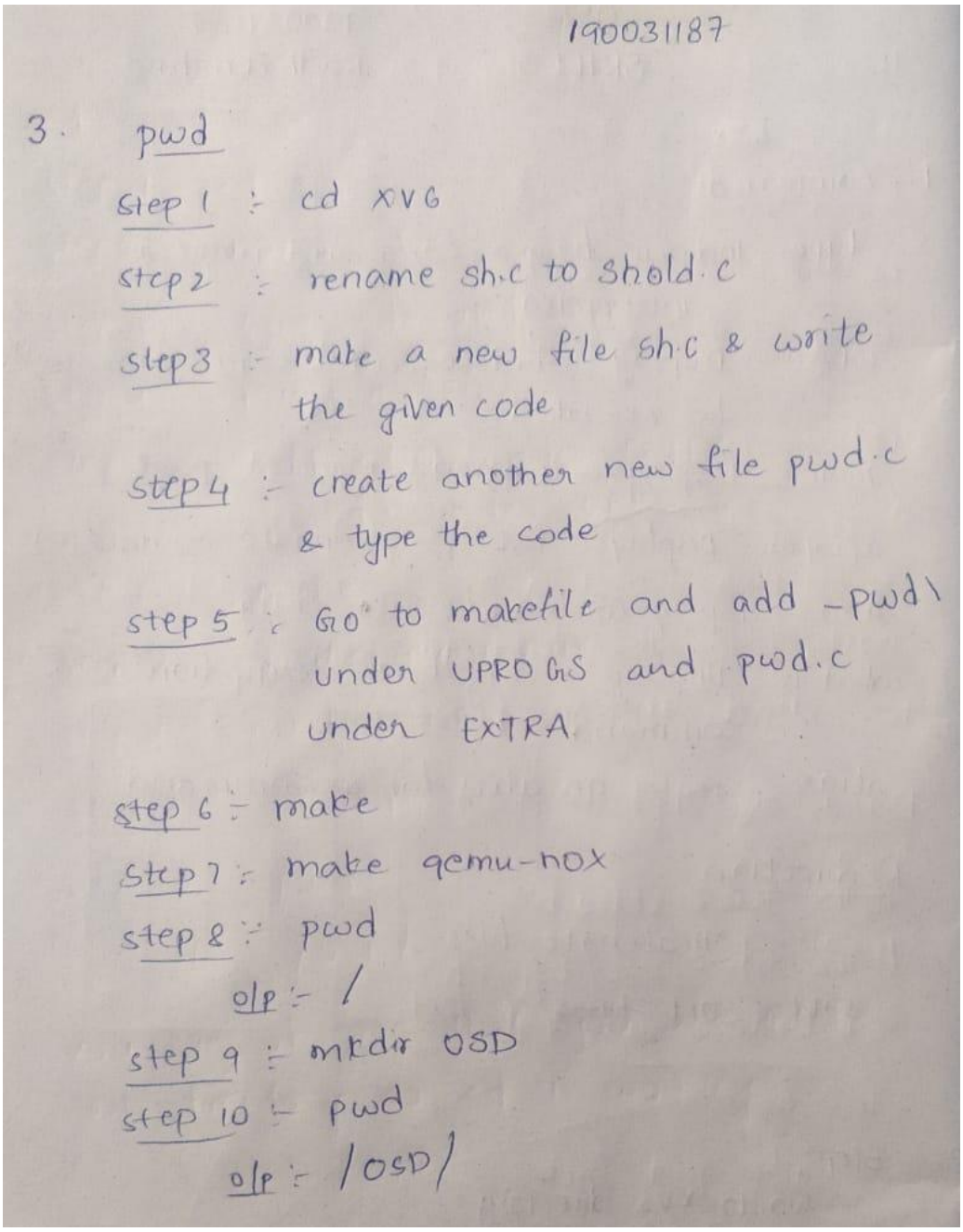

```
#include "types.h"
#include "param.h"
#include "memlayout.h"
#include "mmu.h"
#include "proc.h"
#include "defs.h"
#include "x86.h"
#include "elf.h"
int exec(char *path, char **argv)
{
        char *s, *last; int i, off;
        uint argc, sz, sp, ustack[3+MAXARG+1];
        struct elfhdr elf;
        struct inode *ip;
        struct proghdr ph;
        pde_t *pgdir, *oldpgdir;
        struct proc *curproc = myproc();
        begin_op();
        if((ip = namei(path)) == 0)
        {
                end_op();
                cprintf("exec: fail\n");
                return -1;
        }
```

```
        ilock(ip);
        pgdir = 0;
// Check ELF header
        if(readi(ip, (char*)&elf, 0, sizeof(elf)) != sizeof(elf))
                goto bad;
        if(elf.magic != ELF_MAGIC)
                goto bad;
        if((pgdir = setupkvm()) == 0)
                goto bad;
        // Load program into memory.
        sz = 0;
        for(i=0, off=elf.phoff; i< ph.filesz)
                goto bad;
        sz = 0;
        for(i=0, off=elf.phoff; i< ph.filesz)
                goto bad;
        if(ph.vaddr + ph.memsz < ph.vaddr)
                goto bad;
        if((sz = allocuvm(pgdir, sz, ph.vaddr + ph.memsz)) == 0)
                goto bad;
        if(ph.vaddr % PGSIZE != 0)
                goto bad;
        if(loaduvm(pgdir, (char*)ph.vaddr, ip, ph.off, ph.filesz) < 0)
                goto bad;
        iunlockput(ip);
        end_op();
        ip = 0;
         // Allocate two pages at the next page boundary.
        // Make the first inaccessible. Use the second as the user stack.
        sz = PGROUNDUP(sz);
        if((sz = allocuvm(pgdir, sz, sz + 2*PGSIZE)) == 0)
                goto bad;
```

```
clearpteu(pgdir, (char*)(sz - 2*PGSIZE));
sp = sz;
// Push argument strings, prepare rest of stack in ustack.
for(argc = 0; argv[argc]; argc++)
{
        if(argc >= MAXARG)
                goto bad;
        sp = (sp - (strlen(argv[argc]) + 1)) & ~3;
        if(copyout(pgdir, sp, argv[argc], strlen(argv[argc]) + 1) < 0)
                goto bad;
        ustack[3+argc] = sp;
}
ustack[3+argc] = 0;
ustack[0] = 0xffffffff;
// fake return PC ustack[1] = argc;
ustack[2] = sp - (argc+1)*4;
// argv pointer sp -= (3+argc+1) * 4;
if(copyout(pgdir, sp, ustack, (3+argc+1)*4) < 0)
        goto bad;
// Save program name for debugging.
for(last=s=path; *s; s++)
        if(*s == '/')
                last = s+1;
safestrcpy(curproc->name, last, sizeof(curproc->name));
// Commit to the user image.
oldpgdir = curproc->pgdir; curproc->pgdir = pgdir;
curproc->sz = sz;
curproc->tf->eip = elf.entry;
// main
curproc->tf->esp = sp; switchuvm(curproc);
freevm(oldpgdir);
return 0;
```

```
    bad:
        if(pgdir) freevm(pgdir);
        if(ip){ iunlockput(ip);
        end_op();
        return -1;
}
```

3. pwd, cd, mv(xv6 customization)



190031187

3. pwd

step 1 : cd XV6

step 2 : rename sh.c to shold.c

step 3 : make a new file sh.c & write the given code

step 4 : create another new file pwd.c & type the code

step 5 : Go to makefile and add _pwd\ under UPROGS and pwd.c under EXTRA

step 6 : make

step 7 : make qemu-nox

step 8 : pwd

o/p : /

step 9 : mkdir OSD

step 10 : pwd

o/p : /OSD/

osd-190031187@team-osd:~/xv6

```
SeaBIOS (version 1.11.0-2.el7)


iPXE (http://ipxe.org) 00:03.0 C980 PCI2.10 PnP PMM+1FF94780+1FED4780 C980



Booting from Hard Disk..xv6...
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap 8
init: starting sh
190031187$ pwd
/
190031187$ mkdir OSD
190031187$ cd OSD
190031187$ pwd
/OSD/
190031187$
```

190031187

mv

step 1 : cd xv6

step 2 : create a new file mv.c (vi mv.c)

step 3 : Type the code and save

step 4 : add _mv\ under UPROGS and mv.c under EXTRA in Makefile

step 5 : make

step 6 : make qemu-nox

step 7 : cat >F1.txt

Welcome to OSD course

step 8 : mv F1.txt F2.txt

step 9 : ls

you can observe F1.txt has be renamed as F2.txt.

step 10 : cat F2.txt

Welcome to OSD course

osd-190031187@team-osd:~/xv6

```
Booting from Hard Disk...
xv6...
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap start 58
init: starting sh
190031187$ cat>F1.txt
Wlecome to OSD course
190031187$ mv F1.txt F2.txt
190031187$ cat F2.txt
Wlecome to OSD course
```

190031187

cd

step 1 :- cd xv6

step2 :- nano cd.c

step 3 :- type code and save

step 4 :- nano makefile

step 5 :- add cd.c\ under EXTRA
_cd\ under UPROGS

Step 6 :- make

step 7 :- make qemu-nox

step 8 :- mkdir OSD

step 9 :- cd OSD

step 10 :- pwd

o/p :- /OSD/

you can observe that you are inside OSD directory so cd is working

osd-190031187@team-osd:~/xv6

```
SeaBIOS (version 1.11.0-2.el7)


iPXE (http://ipxe.org) 00:03.0 C980 PCI2.10 PnP PMM+1FF94780+1FED4780 C980



Booting from Hard Disk..xv6...
cpu1: starting 1
cpu0: starting 0
sb: size 1000 nblocks 941 ninodes 200 nlog 30 logstart 2 inodestart 32 bmap 8
init: starting sh
190031187$ pwd
/
190031187$ mkdir OSD
190031187$ cd OSD
190031187$ pwd
/OSD/
190031187$
```